2차시 과제 풀이 (2)

문제 목록

문제 번호	문제 이름	출처
BOJ 20650	Do You Know Your ABCs	USACO 2020 December Bronze 1번
BOJ 18787	Mad Scientist	USACO 2020 February Bronze 2번
BOJ 20647	Cowntagion	USACO 2020 December Silver 1번
BOJ 18780	Timeline	USACO 2020 February Gold 1번
BOJ 18879	The Moo Particle	USACO 2020 US Open Silver 3번
BOJ 20648	Rectangular Pasture	USACO 2020 December Silver 2번
BOJ 20649	Stuck in a Rut	USACO 2020 December Silver 3번
BOJ 18874	Haircut	USACO 2020 US Open Gold 1번

BOJ 20650 Do You Know Your ABCs

가장 큰 값은 $A+B+C$임
두 번째로 큰 값은 $B+C$임
세 번째로 큰 값은 $A+C$임
$(A+B+C)-(B+C)=A$
$(A+B+C)-(A+C)=B$
$(A+B+C)-A-B=C$

BOJ 18787 Mad Scientist

$C_i := [A_i \neq B_i]$라고 정의하면, C에서 연속해서 등장하는 1의 덩어리의 개수를 세는 문제가 된다.

$O(N)$에 해결할 수 있다.

BOJ 20647 Cowntagion

풀이가 잘 생각나지 않는다면 더 쉬운 문제로 바꾼 다음에 풀이를 찾아보는 것이 좋다. 일반적인 트리가 아닌, 성게 그래프(Star Graph)와 선형 트리에서 문제를 풀어보자.

- 성게 그래프의 경우, 소가 N마리 이상이 될 때까지 기다린 다음 한 마리씩 보내는 것이 최적이다. 이때 걸리는 시간은 $N-1+\lceil \log_2 N \rceil$
- 선형 트리에서는 2마리로 만들고 아래로 보내는 것을 반복하는 것이 최적이다. 이때 걸리는 시간은 $2(N-1)$이다.

두 가지 풀이를 조합하면 일반적인 트리에서도 문제를 해결할 수 있다.

트리의 각 정점에서, 소가 (자식 정점의 수 + 1)마리 이상이 될 때까지 기다린 다음, 한 마리씩 보내면 된다.

BOJ 18780 Timeline

먼저 S_i 조건을 무시하고 문제를 풀어보자.

(a, b, x)라는 정보는 b가 a보다 최소 x일 이상 늦게 시작한다는 것을 의미한다. a에서 b로 가는 가중치가 x인 간선을 만들자. (a, b, x) 꼴의 튜플 조건을 모두 만족하는 각 세션의 최소 날짜는 각 정점까지의 **최장 거리**와 동일하다.

DAG에서의 최장 거리 문제로 환원했다면 S_i 조건을 처리하는 방법은 간단하다. 0일차에 열리는 0번 세션을 만들고, 0번 정점에서 i번 정점까지 가는 가중치가 S_i인 간선을 만들면 된다.

$O(N + C)$에 풀 수 있다.

BOJ 18879 The Moo Particle

두 입자 i, j가 상호 반응해서 하나가 없어지는 것을 i번 정점과 j번 정점을 간선으로 연결하는 것이라고 생각하면, 간선을 모두 만들었을 때 컴포넌트의 개수를 세는 것이라고 생각할 수 있다.

좌표의 대소 비교를 하는 문제는 보통 한 축을 기준으로 정렬하고 생각하는 것이 좋다. x 좌표 기준으로 정렬하자.

정렬을 하고 나면, $j \in [1, i-1)$번째 점은 i번째 점과 $y_j \le y_i$인 경우에만 연결될 수 있고, $j \in (i, N]$번째 점은 $y_j \ge y_i$인 경우에만 연결될 수 있다.

그러므로, i번째 점과 $i+1$번째 점이 같은 컴포넌트에 속하지 않을 조건은 $\min(y_1, y_2, \cdots, y_i) > \max(y_{i+1}, y_{i+2}, \cdots, Y_N)$이다. (이 조건을 만족하는 i의 개수) + 1이 문제의 정답이 된다.

BOJ 20648 Rectangular Pasture

2^{1000}과 같이 매우 큰 수는 C++로 처리하기 힘들다. 2^N에서 불가능한 수를 제거하는 것보다는 가능한 경우를 세는 것이 편할 것 같다는 생각을 할 수 있다.

어떤 직사각형 하나를 특정짓기 위해서는 (1) 왼쪽 변의 x좌표, (2) 오른쪽 변의 x좌표, (3) 윗 변의 y좌표, (4) 아랫 변의 y좌표만 알면 된다. x, y좌표를 0부터 10억까지 모두 고려하지 않고, 점이 존재하는 x, y좌표만 고려해도 된다는 것을 알 수 있다. x, y좌표가 각각 서로 다르기 때문에 쉽게 좌표 압축을 해줄 수 있다.

직사각형의 왼쪽 변과 오른쪽 변의 x좌표를 고정하자. 이때 가능한 윗 변과 아랫 변의 조합의 수를 구하면 문제를 해결할 수 있다. 이때 가능한 조합의 수는 (가능한 윗 변의 개수) x (가능한 아랫 변의 개수)이다.

가능한 윗변의 개수는 x좌표가 $x_i \le x \le x_j$이면서 y좌표가 $\max(y_i, y_j) \le y$인 점의 개수와 동일하다. 마찬가지로, 가능한 아랫 변의 개수는 x 좌표가 $x_i \le x \le x_j$이면서 y좌표가 $y \le \min(y_i, y_j)$인 점의 개수와 동일하다.

2차원 영역 상에서 어떤 직사각형 영역에 속하는 점의 개수를 $f(N)$ 시간에 구할 수 있다면 이 문제를 $O(N^2 \cdot f(N))$에 해결할 수 있다. 이러한 작업은 2D Prefix Sum을 이용하면 $O(N^2)$ 전처리를 통해 각 쿼리를 $O(1)$에 해결할 수 있다.

그러므로 전체 시간 복잡도는 $O(N^2)$

BOJ 20649 Stuck in a Rut

Rectangular Pasture문제처럼 x, y좌표가 각각 서로 다르기 때문에 좌표 압축을 생각해볼 수 있다. 이 문제는 **이동 거리**가 중요한 정보이기 때문에 단순하게 좌표 압축을 하면 안 되고, 이동 거리에 대한 정보를 추가적으로 기록해야 한다.

주어진 x좌표를 $\{1, 4, 5, 8, 10\}$이라고 하자. 평소에는 5개로 압축을 하겠지만, 이번에는 9(= 2N-1)개로 압축하고, 압축된 각 덩어리의 가중치를 $\{1, 3, 1, 0, 1, 2, 1, 1, 1\}$로 배정한다. 이렇게 하면 홀수 번째에 소가 배치되고 가중치는 항상 1이며, 이동할 때의 거리는 짝수 번째 칸의 가중치를 통해 조절할 수 있게 된다.

어떤 소 i가 j때문에 직접적으로 Stop 당한다면, 두 소의 이동 경로의 교점에 j가 **더 빨리** 도착했다는 것을 의미한다. 인접한 칸으로 이동하는데 걸리는 시간도 미리 구해놓았으니, 다익스트라 알고리즘을 이용해서 각 소가 어떤 칸에 도착한 시간을 구할 수 있다.

이 문제에서는 직접적으로 Stop 당한 것 뿐만 아니라, 간접적으로 Stop 당한 것도 구해야 한다. 직접적으로 Stop 당한 정보를 이용해 Tree(or Forest)를 구성한 다음, 각 정점을 루트로 하는 서브 트리 안에서 Stop 당한 횟수를 세어주면 된다.

$O(N^2 \log N)$ 시간에 풀 수 있다.

BOJ 18874 Haircut

$0 \le j < N$인 j에 대해, $x < y \wedge A_x > A_y \wedge A_y < j$인 (x, y) 순서쌍의 개수를 구하는 문제다.

먼저, y에 대해 $x < y \wedge A_x > A_y$를 만족하는 x를 구하는 것은 Fenwick Tree를 이용해서 쉽게 처리할 수 있다.

R_j를 $A_y = j$인 y들에 대한 x의 개수라고 정의하면, 문제의 정답은 $\sum_{i=0}^{k-1} R_k$가 된다. $O(N \log N)$에 문제를 풀 수 잇다.